Listing of Claims:

Claims 1-52, where added material is shown in underlined type, deleted material is shown in ~~strikeout type~~:

1. (Currently amended) A method of managing packet queues in a switch having a limited primary memory including a number of queues for switching data packets between input ports and output ports, and connected to a larger secondary memory also including a number of queues, comprising the steps of:

dividing a data stream incoming on the input ports intended for respective output ports into two parts, of which the first part contain flows to be sent to an output port queue of the primary memory and the second part contain flows to be sent to the secondary memory;

wherein the data of the incoming data stream is identified as belonging to flow groups, each flow group containing a number of flows; and

wherein a number of flow groups are assigned to each queue of the primary memory and the secondary memory.

2. (Original) The method according to claim 1, wherein the data of the second part is stored in a third memory before it is sent to the secondary memory.

3. (Original) The method according to claim 2, wherein the primary memory is a fast memory internal on a chip and the secondary memory is external from the chip.

4. (Original) The method according to claim 3, wherein the third memory is provided as store queues forming part of the primary memory.

~~5~~. (Cancelled)

5 ~~6~~. (Currently Amended) The method according to claim ~~5~~ 1, wherein each flow group contains traffic with a specific load value, and the division of the data stream is performed such that a number of flow groups are selected to be sent to said queues of the primary memory in the first part, and the other flow groups are sent to the secondary memory in the second part, the selection being based on the load value, in order to adapt the first part of the data stream to the current capacity of the output port.

6 7. (Original) The method according to claim 6, wherein the load value for each flow group is set to a fixed value.

7 8. (Original) The method according to claim 6, wherein the load value is set by measuring the amount of traffic in the flow groups.

8 9. (Currently amended) The method according to claim ~~5~~ 1, wherein each data packet of the incoming data stream is assigned a hash value based on constant flow information and the flow groups are formed by means of the hash value.

9 10. (Original) The method according to claim 8, wherein the division of the data stream is performed such that a number of flow groups are selected to be sent to said queues of the primary memory in the first part, and the other flow groups are sent to the secondary memory in the second part in order to adapt the first part of the data stream to the current capacity of the output port.

10 11. (Currently amended) The method according to claim ~~5~~ 1, wherein the data packets of the incoming data stream have a priority value and are identified as belonging to priority groups and the flow groups are formed by means of the priority.

11 12. (Currently amended) The method according to claim ~~5~~ 1, wherein the data packets of the incoming data stream have a priority value and are assigned a hash value and the flow groups are formed by means of the priority value and the hash value, each flow group having a certain combination of priority value and hash value.

12 13. (Original) The method according to claims 11 or 12, wherein a number of queues contain flow groups having the same priority value.

13 14. (Currently amended) The method according to claim 11, or 12 ~~or 13~~, wherein the division of the data stream is performed such that priority groups having a priority above a division threshold are sent to said queues of the primary memory in the first part, while priority groups having a priority below said threshold are sent to the secondary memory in the second part.

15. (Cancelled)

14 16. (Original) The method according to claim 1, wherein the division of the data stream is performed, so that the total load of the flows of the first part is lesser than or equal to the total output capacity of the output ports.

15 17. (Original) The method according to claim 14 16, wherein the total output capacity of the output ports is set to a fixed value.

16 18. (Original) The method according to claim 14 16, wherein the total output capacity of the output ports is set by measuring the traffic passing the output ports.

A2 17 19. (Original) The method according to claim 1, wherein a scheduler selects packets from the primary memory and the secondary memory.

18 20. (Original) The method according to claim 17 19, wherein the scheduler first selects packets from the primary memory, then, if the primary memory is empty, the scheduler selects packets from the secondary memory.

19 21. (Original) The method according to claim 17 19, wherein the data packets have a priority value, and the scheduler selects packets on a strict priority basis from the primary memory and the secondary memory, and if packets have the same priority, packets from the primary memory are selected first.

20 22. (Original) The method according to claim 19 21, wherein the output ports share the same bandwidth from the secondary memory, and, when the whole bandwidth is occupied by the other output ports, as seen from one output port, then, the scheduler is able to read from the primary memory, even though the priority order may be broken.

21 23. (Original) The method according to claim 2, wherein flows are integrated back from the secondary memory to the primary memory, by means of the following steps: the flow in the relevant group to the secondary memory is blocked and stored in the third memory, and the queue of the secondary memory is emptied; when this is done, the contents of the third memory is

moved to the internal queue of the primary memory and the relevant flow is switched to the first part.

22 24. (Original) The method according to claim 23, wherein the integration process only starts if the lengths of the respective queues of the secondary memory and the third memory are smaller than predetermined values.

23 25. (Original) The method according to claim 23, wherein the integration process is interrupted, if the length of the respective queue of the secondary memory rises above a certain value by releasing the blocking in the third memory and sending on the flow to the secondary memory.

24 26. (Original) The method according to claim 1, wherein at least one flow in the first part is moved to the second part, if the load of the flows currently located in the first part of the incoming data stream exceeds the capacity of the output ports.

25 27. (Currently amended) An arrangement for managing packet queues in a switch having a limited primary memory including a number of queues for switching data packets between input ports and output ports, and connected to a larger secondary memory also including a number of queues, comprising:

means for dividing a data stream incoming on the input ports intended for respective output ports into two parts, of which the first part contain flows to be sent to an output port queue of the primary memory and the second part contain flows to be sent to the secondary memory;

wherein the data of the incoming data stream is identified as belonging to flow groups, each flow group containing a number of flows; and

wherein a number of flow groups are assigned to each queue of the primary memory and the secondary memory.

26 28. (Original) The arrangement according to claim 27, wherein the data of the second part is stored in a third memory before it is sent to the secondary memory.

27 29. (Original) The arrangement according to claim 28, wherein the primary memory is a fast memory internal on a chip and the secondary memory is external from the chip.

28 30. (Original) The arrangement according to claim 29, wherein the third memory is provided as store queues forming part of the primary memory. 27

31. (Cancelled)

29 32. (Currently Amended) The arrangement according to claim 31 27, wherein each flow group contains traffic with a specific load value, and the division of the data stream is performed such that a number of flow groups are selected to be sent to said queues of the primary memory in the first part, and the other flow groups are sent to the secondary memory in the second part, the selection being based on the load value, in order to adapt the first part of the data stream to the current capacity of the output port.

30 33. (Original) The arrangement according to claim 32, wherein the load value for each flow group is set to a fixed value.

31 34. (Original) The arrangement according to claim 32, wherein the load value is set by measuring the amount of traffic in the flow groups.

32 35. (Currently amended) The arrangement according to claim 31 27, wherein each data packet of the incoming data stream is assigned a hash value based on constant flow information and the flow groups are formed by means of the hash value.

33 36. (Original) The arrangement according to claim 35, wherein the division of the data stream is performed such that a number of flow groups are selected to be sent to said queues of the primary memory in the first part, and the other flow groups are sent to the secondary memory in the second part in order to adapt the first part of the data stream to the current capacity of the output port.

34 37. (Currently amended) The arrangement according to claim 31 27, wherein the data packets of the incoming data stream have a priority value and are identified as belonging to priority groups and the flow groups are formed by means of the priority.

35 ~~38~~. (Currently amended) The arrangement according to claim ~~31~~ ~~27~~ 25, wherein the data packets of the incoming data stream have a priority value and are assigned a hash value and the flow groups are formed by means of the priority value and the hash value, each flow group having a certain combination of priority value and hash value.

36 ~~39~~. (Original) The arrangement according to claim ~~37~~ 34 or ~~38~~ 35, wherein a number of queues contain flow groups having the same priority value.

37 ~~40~~. (Currently Amended) The arrangement according to claim ~~37~~ 34, or ~~38~~ 35 ~~or 39~~, wherein the division of the data stream is performed such that priority groups having a priority above a division threshold are sent to said queues of the primary memory in the first part, while priority groups having a priority below said threshold are sent to the secondary memory in the second part.

~~41~~. (Cancelled)

G2

38 ~~42~~. (Original) The arrangement according to claim ~~27~~ 25, wherein the division of the data stream is performed, so that the total load of the flows of the first part is lesser than or equal to the total output capacity of the output ports.

39 ~~43~~. (Original) The arrangement according to claim ~~42~~ 38, wherein the total output capacity of the output ports is set to a fixed value.

40 ~~44~~. (Original) The arrangement according to claim ~~42~~ 38, wherein the total output capacity of the output ports is set by measuring the traffic passing the output ports.

41 ~~45~~. (Original) The arrangement according to claim ~~27~~ 25, wherein a scheduler selects packets from the primary memory and the secondary memory.

42 ~~46~~. (Original) The arrangement according to claim ~~45~~ 41, wherein the scheduler first selects packets from the primary memory, then, if the primary memory is empty, the scheduler selects packets from the secondary memory.

43 47. (Original) The arrangement according to claim 45, 41 wherein the data packets have a priority value, and the scheduler selects packets on a strict priority basis from the primary memory and the secondary memory, and if packets have the same priority, packets from the primary memory are selected first.

44 48. (Original) The arrangement according to claim 47, 43 wherein the output ports share the same bandwidth from the secondary memory, and, when the whole bandwidth is occupied by the other output ports, as seen from one output port, then, the scheduler is able to read from the primary memory, even though the priority order may be broken.

45 49. (Original) The arrangement according to claim 28, 26 wherein flows are integrated back from the secondary memory to the primary memory, by means of the following steps: the flow in the relevant group to the secondary memory is blocked and stored in the third memory, and the queue of the secondary memory is emptied; when this is done, the contents of the third memory is moved to the internal queue of the primary memory and the relevant flow is switched to the first part.

G2

46 50. (Original) The arrangement according to claim 49, 45 wherein the integration process only starts if the lengths of the respective queues of the secondary memory and the third memory are smaller than predetermined values.

47 51. (Original) The arrangement according to claim 49, 45 wherein the integration process is interrupted, if the length of the respective queue of the secondary memory rises above a certain value by releasing the blocking in the third memory and sending on the flow to the secondary memory.

48 52. (Original) The arrangement according to claim 27, 25 wherein at least one flow in the first part is moved to the second part, if the load of the flows currently located in the first part of the incoming data stream exceeds the capacity of the output ports.